HRM Time Stamps

Interim scheme for events

Fri, Dec 20, 2002

It will be necessary for the HRM hardware to supply a time stamp for a selected event that can be used to determine an offset between the time stamps obtained from the digital PMC board and those used for the slow data in the HRM. This note describes a scheme for obtaining such an offset that can serve temporarily until the hardware solution is in place.

Event time stamps are captured by the digital PMC board. During the clock event interrupt, they are placed into the event times array in low memory. Slow Data time stamps are stored by the HRM hardware into an array in HRM memory. We need to compare these two time stamps in order to derive an offset that can be used to relate one to the other.

Assume that the following logic is executed in the `Update` task during every 15 Hz cycle. Read the current digital PMC time stamp by accessing the two 16-bit registers that hold it. There must be three accesses to do this correctly. Read the hi word, read the lo word, then read the hi word again to be sure it has not changed, else repeat.

Next read the current Slow Data time stamp. Reading the block counter register, back up by one, access the time stamp found in the HRM Slow Data time stamp array, and add 150 to it. The reason to back up one is that the block number register is incremented just before the corresponding time stamp is stored, so this is a way to be sure the time stamp sampled is stable. Adding 100 microseconds gets the current time stamp. It will be stable for 100 microseconds, so 50 is added.

The offset result is defined as (`evts – sdts`), where `evts` is the event time stamp and `sdts` is the Slow Data time stamp just determined. This result, of course, is only accurate within 100 microseconds. But since the 100 microsecond ticks occur asynchronously with the 15 Hz events, one could use an averaging scheme to improve the accuracy of this offset. One might average over 100 offsets, say, to get a more accurate result. The first such average could be obtained in about 7 seconds.

Note that the above scheme does not require adding to the event interrupt code. This logic can be executed any time within the cycle that is convenient. It may be easiest to think about if it is done synchronously with 15 Hz. It may be useful to include some diagnostics that show the departure from the average of each determined offset. A log of successive average offset results may be useful, too, or perhaps differences between successive average offsets.

Armed with the offset value, during `RFTData` processing, when a time stamp is obtained for each data point, the offset value should be added before it is used to relate to clock event time stamps. The offset addition provides time stamps that are properly comparable to those found in the event time stamp array.

In case more than one Slow Data module is used within a system, one must determine offsets for each one, since the time stamps coming from the Slow Data modules are independent.

If the Slow Data is not triggered at 100 microsecond intervals, the logic above is not quite correct. But as an interim solution to the event relevant time stamp problem, we can assume 10 KHz is used.

This interim scheme is somewhat crude, but it may be good enough to be useful in plotting data based upon clock events. The listype 82 support returns time stamps in 10 microsecond units. The `FTPMAN` continuous time stamps are returned in 100 microsecond units. The accuracy of the interim scheme will reasonably match the resolution of the time stamps that are provided for a requester.

For more on this topic, see also the note called *HRM Time Stamp Management*.